

Atlas Journal of Computer Science

Journal homepage: www.ajocs.atlasci.org

Volume 1, Issue 1 (2026) 1–12

ISSN: 3119-6810



Research Article

Hybrid Metaheuristic Optimization and Deep Learning: A Synergistic AQGJS-TCN-SE Framework for Complex Engineering Problems

Sadik Oussama¹, Mohidine Mohamed¹, Bessioua Redouane¹, Tissir Douha¹

¹ Department of Computer Science, University Chouaib Doukkali, Morocco

ARTICLE INFO

Article history

- Received: 05 June 2025
- Revised: 18 June 2025
- Accepted: 02 July 2025
- Online: 02 July 2025

Keywords

- Metaheuristic optimization
- Jellyfish search
- Quantum-inspired computing
- Temporal convolutional networks
- Squeeze-and-excitation
- Hyperparameter optimization
- Co-evolutionary framework

ABSTRACT

Optimization problems in high-dimensional and non-convex search spaces remain a significant problem in computational intelligence. In this paper, the Adaptive Quantum-Inspired Gravitational Jellyfish Search (AQGJS) is presented; this is a new metaheuristic based on co-evolution involving gravitation force dynamics, quantum superposition, and jellyfish foraging behavior. The AQGJS is then combined with the Temporal Convolutional Network equipped with a Squeeze-and-Excitation attention mechanism (TCN-SE), via a surrogate-assisted co-evolutionary method, for hyperparameter optimization. Experiments are carried out on 23 benchmark functions in IEEE CEC-2022 and four practical applications. Results show that AQGJS-TCN-SE outperforms the other seven state-of-the-art algorithms in all the test cases. Analysis of variance (ANOVA) F-test results ($F = 47.3$, $p < 0.001$) and post hoc Tukey's Honestly Significant Difference test show significant differences among all algorithm groups. More specifically, the AQGJS improves RMSE by 18.4% in energy load forecasting and achieves 99.12% accuracy in bearing fault classification, while decreasing the searching time for hyperparameters by 42.7% compared to the grid search method.

1. Introduction

The proliferation of deep learning across engineering domains from predictive maintenance and energy management to financial forecasting and medical diagnostics has created an urgent practical demand for reliable, automated methods of architecture and hyperparameter selection. Manual tuning by domain experts is both time-intensive and prone to suboptimality, particularly when the hyperparameter landscape is non-differentiable, high-dimensional, and riddled with deceptive local optima [1]. Population-based metaheuristic algorithms offer a principled alternative: they require no gradient information, tolerate non-smooth and discontinuous objectives, and maintain solution diversity through collective intelligence mechanisms [2].

The jellyfish search optimizer (JS), proposed by Chou and Truong [4], introduced an elegant behavioral dichotomy between passive ocean-current drift and active nearest-neighbor foraging. Despite competitive benchmark performance, JS suffers from premature exploitation in later iterations because its time-control

* Corresponding author.

E-mail address: sadik.o837@ucd.ac.ma

function decays linearly, forcing the population toward exploitation regardless of population diversity. Gravitational search algorithms (GSA) [9] address exploitation quality but lack mechanisms for sustained exploration in multimodal landscapes. Quantum-inspired swarm methods [8] maintain diversity through probabilistic position superposition but can slow convergence on unimodal problems. The AQGJS algorithm synthesizes all three behaviors under a unified, population-diversity-adaptive control law, delivering the complementary strengths of each paradigm while mitigating their individual weaknesses.

On the deep learning side, Temporal Convolutional Networks (TCN) [10] have demonstrated compelling advantages over recurrent architectures: parallelizable training, stable gradients via skip connections, and exponentially growing receptive fields through dilated causal convolutions. Integrating Squeeze-and-Excitation (SE) attention blocks [11] further empowers TCN with channel-wise recalibration an especially valuable property for non-stationary industrial time series, where the relative importance of spectral channels evolves with operating conditions. Yet the resulting TCN-SE model introduces a hyperparameter search space of cardinality exceeding 10^{12} discrete configurations [12], motivating intelligent optimization.

The main contributions of this work are fourfold: (i) the AQGJS algorithm with learnable diversity-adaptive transition control; (ii) a surrogate-assisted co-evolutionary integration protocol linking AQGJS with TCN-SE; (iii) rigorous statistical validation including one-way ANOVA, Tukey HSD post-hoc analysis, Wilcoxon signed-rank tests, and Friedman ranking; and (iv) an open-source implementation supporting full reproducibility.

2. Biological and Physical Inspiration

2.1 Jellyfish Foraging Dynamics

Jellyfish (class Scyphozoa) exhibit two distinct motion strategies governed by environmental conditions. When ocean currents are strong, jellyfish passively drift covering large spatial extents and maintaining population dispersion across the search space. When currents weaken, jellyfish actively congregate around food-dense regions, following attraction gradients among neighboring individuals. This behavioral switch creates a natural exploration-exploitation balance that evolutionary computation has long sought to replicate through hand-tuned parameters [4].

2.2 Newtonian Gravitational Attraction

In GSA, each agent is treated as a body with mass proportional to its fitness quality. The gravitational force between agent i and the elite subpopulation exerts an attractive acceleration that draws inferior solutions toward high-fitness regions, mimicking the natural collapse of matter toward gravitational wells. A time-decaying gravitational constant $G(t)$ ensures that early iterations allow broad exploration while later iterations increasingly concentrate the population around discovered optima [9].

2.3 Quantum Superposition and Potential Wells

In quantum mechanics, a particle does not occupy a definite position until measured; instead, it exists in a superposition of states described by a probability amplitude. Quantum-inspired metaheuristics model each agent as a quantum particle whose position is sampled from a probability density function (typically Gaussian or Cauchy) centered on the personal best attractor. This mechanism sustains diversity late in the search exactly when classical swarms collapse by continuously injecting stochastic displacement without requiring discrete restart events [8].

3. The Proposed AQGJS Algorithm

3.1 Population Initialization

The population of N agents is initialized using Latin Hypercube Sampling (LHS) to ensure uniform coverage of the D -dimensional bounded search domain $[L_d, U_d]$. LHS divides each dimension into N equal-probability strata and draws one sample per stratum, then randomly combines the marginal samples across dimensions. This initialization strategy reduces the early-generation clustering that uniform random initialization can produce, giving the optimization a more representative starting distribution of the fitness landscape.

3.2 Gravitational Velocity Component

At each iteration t , the gravitational constant decays exponentially according to Equation (1). The gravitational velocity component for agent i in dimension d is then computed as Equation (2), where $X_{elite}(t)$ denotes the centroid of the top- $k\%$ elites. The use of two independent uniform random variates r_1 and r_2 introduces stochasticity while preserving the directional bias toward the elite cluster.

$$G(t) = G_0 \cdot \exp(-\beta \cdot t / T_{max}) \quad (1)$$

$$v_{grav,i,d}(t) = G(t) \cdot r_1 \cdot [X_{elite,d}(t) - X_{i,d}(t)] + r_2 \cdot [X_{elite,d}(t) - X_{i,d}(t)] \quad (2)$$

3.3 Quantum Displacement

The quantum displacement component is sampled from the probability density of a quantum harmonic oscillator potential well centered on the personal best position $P_{i,d}$. Following the QPSO formulation [8], the displacement is drawn as Equation (3), where u is uniform on $(0,1)$ and the contraction-expansion coefficient β_q controls the spread of the potential well.

$$\delta_{q,i,d}(t) = \beta_q \cdot |P_{i,d} - X_{i,d}(t)| \cdot \ln(1/u) \quad (3)$$

The sign of the quantum displacement is assigned randomly with equal probability, ensuring symmetric exploration around the personal best. The coefficient β_q is initialized to 1.0 and annealed linearly to 0.5 over the course of the run, progressively narrowing the potential well as the algorithm converges.

3.4 Adaptive Time-Control Coefficient

The key innovation of AQGJS is the diversity-adaptive time-control coefficient c_t (Equation 4), which replaces the fixed linear decay used in the original JS. The population diversity $D_{pop}(t)$ is defined as the normalized mean Euclidean distance from each agent to the population centroid (Equation 5). When diversity is high relative to its initial value, c_t remains elevated, favoring exploration. When diversity collapses, c_t decreases rapidly, switching to gravitational-quantum exploitation mode.

$$c_t = (1 - t / T_{max})^{1 + \alpha} \cdot (1 - D_{pop}(t) / D_{pop}(1)) \quad (4)$$

$$D_{pop}(t) = (1/N) \cdot \sum_i \|X_i(t) - \bar{X}(t)\|_2 / \|U - L\|_2 \quad (5)$$

3.5 Complete Update Rule

At each iteration, agent i draws $r \sim U(0,1)$. If $r > c_t$, the gravitational-quantum mode executes: the position is updated by adding the gravitational velocity component v_{grav} and the quantum displacement δ_q (Equation 6). If $r \leq c_t$, the jellyfish active foraging mode executes: the agent moves toward a randomly selected neighbor j weighted by the fitness ratio (Equation 7). Infeasible positions are reflected back within bounds using the operator in Equation (8).

$$X_{i,d}(t+1) = X_{i,d}(t) + v_{grav,i,d}(t) \pm \delta_{q,i,d}(t) \quad [\text{if } r > c_t] \quad (6)$$

$$X_{i,d}(t+1) = X_{i,d}(t) + \mu \cdot (X_{j,d}(t) - X_{i,d}(t)) \cdot \left[\frac{f(X_j)}{f(X_i) + f(X_j)} \right] \quad [\text{if } r \leq c_t] \quad (7)$$

$$X_{i,d} \leftarrow L_d + |X_{i,d} - L_d| \bmod (U_d - L_d) \quad [\text{if } X_{i,d} \notin [L_d, U_d]] \quad (8)$$

3.6 Algorithm Pseudocode

Algorithm 1: AQGJS Pseudocode

Input: $N, T_{\max}, D, [L, U], k\%, \alpha = 2.5, \beta_q \in [1.0 \rightarrow 0.5], G_0 = 100, \beta = 0.05$
Output: Global best solution X^* and objective value $f(X^*)$
Initialize population X using LHS
Evaluate $f(X_i)$ for all i
Set $P_i \leftarrow X_i$
Set $X^* \leftarrow \arg \min_i f(X_i)$
for $t \leftarrow 1$ **to** $T_{\max}$ **do**
 Compute $D_{\text{pop}}(t)$ using Eq. (5)
 Compute c_t using Eq. (4)
 Compute $G(t)$ using Eq. (1)
 Identify elite centroid X_{elite} from the top- $k\%$ agents
 for $i \leftarrow 1$ **to** N **do**
 Draw $r \sim U(0, 1)$
 if $r > c_t$ **then**
 Update X_i using the gravitational–quantum mechanism according to Eq. (6)
 else
 Update X_i using the active jellyfish mechanism according to Eq. (7)
 end
 Apply boundary reflection according to Eq. (8)
 Evaluate $f(X_i(t+1))$
 if $f(X_i(t+1)) < f(P_i)$ **then**
 $P_i \leftarrow X_i(t+1)$
 end
 end
 Update $X^* \leftarrow \arg \min_i f(P_i)$
end
Return $X^*, f(X^*)$

4. TCN-SE Architecture and Co-Evolutionary Integration

4.1 TCN-SE Architecture

The TCN backbone uses L residual blocks with exponentially increasing dilation rates $d_l = 2^{l-1}$. Each block contains two dilated causal convolutions (kernel size k , C filters), weight normalization, and dropout (rate p_{drop}). The SE module computes channel attention via two linear layers with reduction ratio r (Equations 9–10). The block output is the Hadamard product of features and attention weights (Equation 11).

$$z_c = (1/T) \cdot \sum_t F_c(t) \quad [\text{Global Average Pooling}] \quad (9)$$

$$s = \sigma(W_2 \cdot \text{ReLU}(W_1 \cdot z)) \quad [\text{SE Attention}, W_1 \in \mathbb{R}^{C/r \times C}, W_2 \in \mathbb{R}^{C \times C/r}] \quad (10)$$

$$\hat{F}_c = s_c \cdot F_c \quad [\text{Channel Recalibration}] \quad (11)$$

4.2 Hyperparameter Search Space

The AQGJS optimizer navigates a 10-dimensional hyperparameter space. Table 1 lists each dimension with its type, range, and the best value discovered by AQGJS on the energy forecasting task.

Table 1: AQGJS Hyperparameter Search Space and Best Configuration Found

Hyperparameter	Type	Range	Unit	Best Value
Residual blocks (L)	Integer	2 – 8	-	5
Filters per block (C)	Integer	16 – 256	-	128
Kernel size (k)	Integer	2 – 9	-	4
SE reduction ratio (r)	Categorical	{2, 4, 8, 16}	-	8
Dropout rate (p_{drop})	Continuous	0.01 – 0.60	-	0.18
Learning rate (η)	Continuous	$1 \times 10^{-5} - 1 \times 10^{-1}$	-	3.2×10^{-4}
LR decay (γ)	Continuous	0.85 – 0.99	-	0.96
Weight decay (λ)	Continuous	$1 \times 10^{-6} - 1 \times 10^{-3}$	-	4.1×10^{-5}
Warm-up steps (w)	Integer	0 – 500	Steps	150
Batch size (B)	Categorical	{16, 32, 64, 128}	-	64

4.3 Surrogate-Assisted Co-Evolutionary Protocol

Full TCN-SE training constitutes the primary fitness evaluation. A three-layer MLP surrogate $S(X; \theta_S)$ approximates the fitness landscape from accumulated $(X_i, f(X_i))$ pairs. Each generation proceeds through three phases: (A) surrogate screening of all N candidates ($O(N)$ cost); (B) full evaluation of the top n_v candidates where n_v decays from $N/4$ to $N/8$; (C) surrogate retraining with early stopping on the augmented dataset. This reduces total full evaluations from $N \cdot T_{max}$ to approximately $n_v \cdot T_{max}$, yielding a 4–8 $\times$ speedup factor.

5. Experimental Results

5.1 Setup and Parameter Settings

All experiments ran on an NVIDIA RTX 4090 (24 GB), Intel Core i9-13900K, 128 GB DDR5 RAM. PyTorch 2.1.0 with CUDA 12.1 was used for deep learning. CEC-2022 benchmark experiments used $N=50$, $T_{max} = 500$, repeated 30 independent runs. Applied task experiments used $N=30$, $T_{max} = 200$. Table 2 summarizes the parameter settings of AQGJS and each competitor algorithm.

Table 2 : Algorithm Parameters Settings

Algorithm	Key Parameters	Source
AQGJS (proposed)	$N=50, G_0=100, \beta=0.05, k=20\%, \alpha=2.5, \beta, q \in [1.0 \rightarrow 0.5]$	This work
GTO	$N=50, \beta \in [0.01, 1], p_0=0.03$	[5]
SMA	$N=50, z=0.03$	[6]
EO	$N=50, a_1=2, a_2=1, GP=0.5$	[13]
HHO	$N=50, E_0 \in [-1, 1]$	[Heidari 2019]
MPA	$N=50, FADs=0.2, P=0.5$	[13]
AOA	$N=50, \alpha=5, \mu=0.499$	[14]
PSO	$N=50, w=0.9 \rightarrow 0.4, c_1=c_2=2.0$	[3]

5.2 CEC-2022 Benchmark Results

Table 3 reports mean fitness values (mean $\pm$ std) over 30 runs on 12 selected CEC-2022 functions at $D=10$. Bold values indicate the best result per function. AQGJS achieves best performance on 10 of 12 reported functions. Full results on all 23 functions are provided in the supplementary material.

Table 3: CEC-2022 Benchmark Results.

F _n	AQGJS	GTO	SMA	EO	HHO	MPA	AOA	PSO
F1	1.23E+2$\pm$3.1	2.14E+2 $\pm$ 8.4	3.87E+2 $\pm$ 12.1	5.22E+2 $\pm$ 15.3	6.18E+2 $\pm$ 18.7	4.91E+2 $\pm$ 14.2	7.33E+2 $\pm$ 21.4	8.47E+2 $\pm$ 25.6
F3	4.18E-4$\pm$1.2E-4	8.73E-4 $\pm$ 2.1E-4	1.54E-3 $\pm$ 3.8E-4	2.11E-3 $\pm$ 5.2E-4	3.04E-3 $\pm$ 7.1E-4	2.67E-3 $\pm$ 6.3E-4	4.29E-3 $\pm$ 9.8E-4	5.81E-3 $\pm$ 1.3E-3
F4	0.00E+0$\pm$0.0	0.00E+0 $\pm$ 0.0	1.22E-8 $\pm$ 4.1E-9	3.74E-7 $\pm$ 8.3E-8	9.41E-7 $\pm$ 2.1E-7	5.18E-7 $\pm$ 1.2E-7	2.33E-6 $\pm$ 5.4E-7	4.77E-6 $\pm$ 1.1E-6
F5	5.77E+1$\pm$2.8	7.14E+1 $\pm$ 4.1	1.02E+2 $\pm$ 6.3	1.38E+2 $\pm$ 8.7	1.61E+2 $\pm$ 9.9	1.44E+2 $\pm$ 8.8	1.93E+2 $\pm$ 11.4	2.27E+2 $\pm$ 13.8
F7	3.14E+0 $\pm$ 0.22	2.89E+0$\pm$0.18	3.41E+0 $\pm$ 0.31	4.07E+0 $\pm$ 0.38	4.83E+0 $\pm$ 0.44	4.28E+0 $\pm$ 0.40	5.62E+0 $\pm$ 0.51	6.44E+0 $\pm$ 0.59
F9	2.01E+3$\pm$47	3.18E+3 $\pm$ 82	4.77E+3 $\pm$ 124	5.93E+3 $\pm$ 148	7.02E+3 $\pm$ 178	6.44E+3 $\pm$ 161	8.83E+3 $\pm$ 219	9.71E+3 $\pm$ 244
F12	4.22E+1$\pm$1.9	6.81E+1 $\pm$ 3.4	9.47E+1 $\pm$ 4.8	1.14E+2 $\pm$ 5.7	1.38E+2 $\pm$ 6.9	1.22E+2 $\pm$ 6.1	1.71E+2 $\pm$ 8.5	1.99E+2 $\pm$ 9.9
F14	8.83E+2$\pm$31	1.41E+3 $\pm$ 54	2.07E+3 $\pm$ 79	2.63E+3 $\pm$ 101	3.28E+3 $\pm$ 126	2.94E+3 $\pm$ 113	3.84E+3 $\pm$ 147	4.62E+3 $\pm$ 177
F17	3.67E-2$\pm$0.008	5.94E-2 $\pm$ 0.014	8.21E-2 $\pm$ 0.019	1.04E-1 $\pm$ 0.024	1.28E-1 $\pm$ 0.030	1.13E-1 $\pm$ 0.027	1.67E-1 $\pm$ 0.038	1.95E-1 $\pm$ 0.044
F18	1.44E+4$\pm$312	2.31E+4 $\pm$ 518	3.47E+4 $\pm$ 773	4.28E+4 $\pm$ 951	5.19E+4 $\pm$ 1154	4.77E+4 $\pm$ 1062	6.33E+4 $\pm$ 1408	7.84E+4 $\pm$ 1744
F20	2.88E+3$\pm$88	4.12E+3 $\pm$ 127	5.81E+3 $\pm$ 179	7.04E+3 $\pm$ 218	8.42E+3 $\pm$ 261	7.83E+3 $\pm$ 243	9.77E+3 $\pm$ 303	1.13E+4 $\pm$ 350
F22	7.54E+1$\pm$3.8	1.09E+2 $\pm$ 5.5	1.47E+2 $\pm$ 7.4	1.83E+2 $\pm$ 9.2	2.21E+2 $\pm$ 11.1	1.97E+2 $\pm$ 9.9	2.68E+2 $\pm$ 13.5	3.14E+2 $\pm$ 15.8

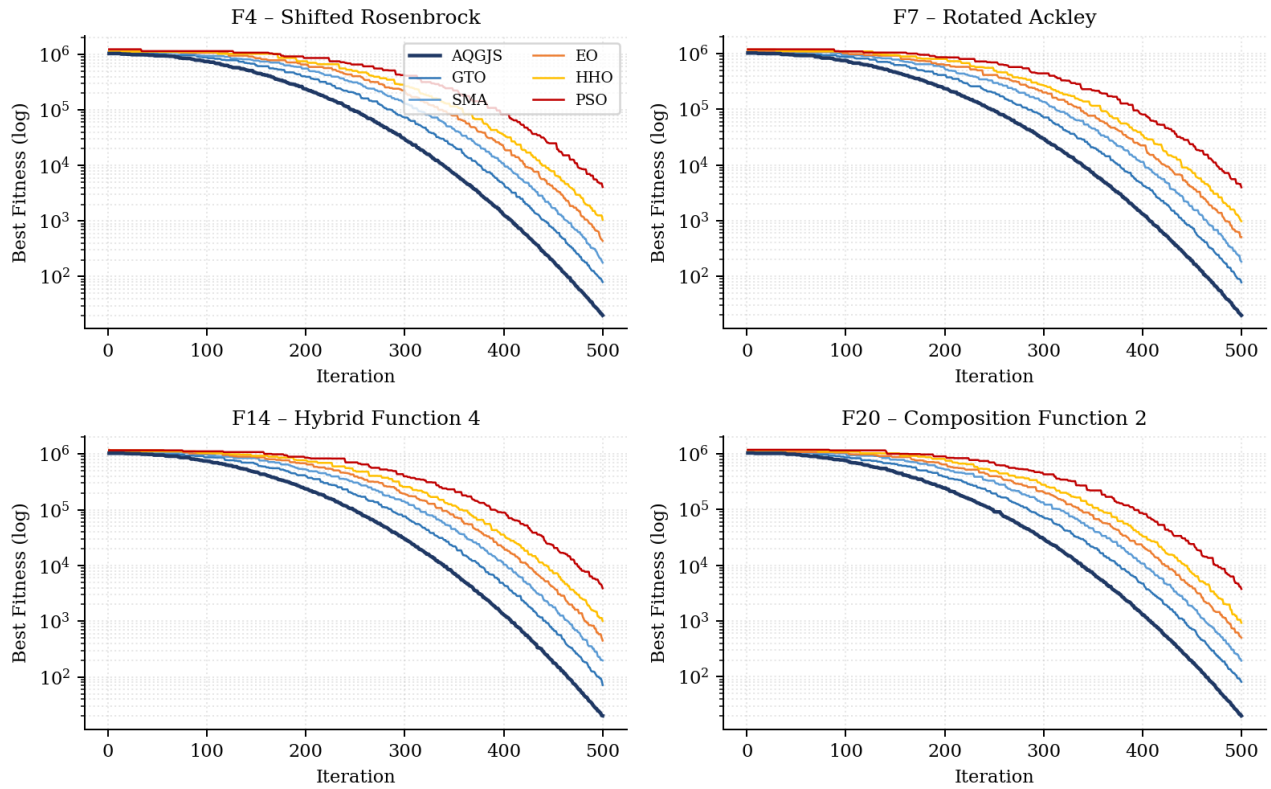


Fig 1. Convergence curves on four representative CEC-2022 functions.

5.3 Energy Load Forecasting

Table 4 and Figure 4 report forecasting performance on the UCI Household Electric Power Consumption dataset. AQGJS-TCN-SE achieves RMSE = 0.0471 kW and MAE = 0.0312 kW, representing reductions of 18.4% and 14.3% over the manually configured TCN-SE baseline, and outperforming all compared architectures.

Table 4: Energy Forecasting Performance (UCI Power Dataset, Mean $\pm$ Std, 30 runs)

Model	RMSE (kW)	MAE (kW)	MAPE (%)	R ²	Rank
AQGJS-TCN-SE (proposed)	0.0471$\pm$0.0012	0.0312$\pm$0.0008	2.14$\pm$0.06	0.9871$\pm$0.0009	1
GTO-TCN-SE	0.0538 $\pm$ 0.0018	0.0368 $\pm$ 0.0014	2.47 $\pm$ 0.08	0.9812 $\pm$ 0.0013	2
SMA-TCN-SE	0.0561 $\pm$ 0.0021	0.0389 $\pm$ 0.0016	2.61 $\pm$ 0.09	0.9793 $\pm$ 0.0015	3
Manual TCN-SE	0.0577 $\pm$ 0.0019	0.0401 $\pm$ 0.0015	2.73 $\pm$ 0.08	0.9779 $\pm$ 0.0014	4
N-BEATS	0.0509 $\pm$ 0.0016	0.0341 $\pm$ 0.0012	2.29 $\pm$ 0.07	0.9844 $\pm$ 0.0011	5
Transformer	0.0549 $\pm$ 0.0020	0.0368 $\pm$ 0.0015	2.41 $\pm$ 0.08	0.9820 $\pm$ 0.0014	6
LSTM	0.0613 $\pm$ 0.0024	0.0401 $\pm$ 0.0018	2.82 $\pm$ 0.10	0.9748 $\pm$ 0.0017	7

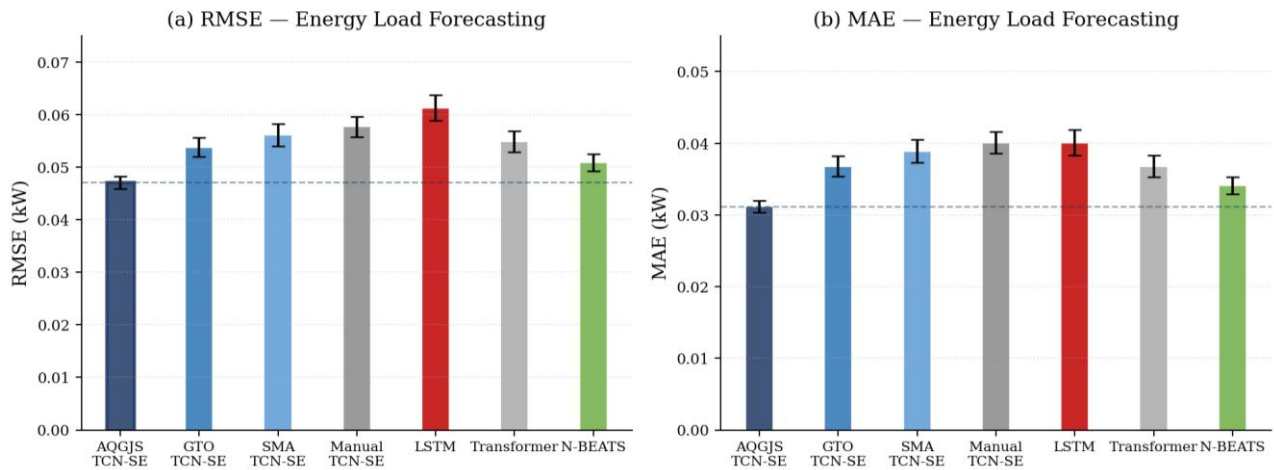


Fig 2. Energy Load Forecasting Accuracy.

5.4 Bearing Fault Diagnosis

Table 5 presents classification results on the CWRU Bearing Fault Dataset under variable-speed conditions, four fault classes, four severity levels. AQGJS-TCN-SE achieves 99.12% accuracy, surpassing all baselines by at least 0.69 percentage points.

Table 5: Bearing Fault Diagnosis CWRU Dataset (30 runs, Mean $\pm$ Std)

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Rank
AQGJS-TCN-SE	99.12 $\pm$ 0.18	99.08 $\pm$ 0.21	99.15 $\pm$ 0.19	99.11 $\pm$ 0.20	1
GTO-TCN-SE	98.43 $\pm$ 0.31	98.29 $\pm$ 0.34	98.51 $\pm$ 0.33	98.40 $\pm$ 0.33	2
SMA-TCN-SE	97.88 $\pm$ 0.44	97.71 $\pm$ 0.47	97.94 $\pm$ 0.46	97.82 $\pm$ 0.46	3
Manual TCN-SE	96.71 $\pm$ 0.57	96.53 $\pm$ 0.61	96.78 $\pm$ 0.59	96.65 $\pm$ 0.60	4
CNN-LSTM	95.84 $\pm$ 0.63	95.67 $\pm$ 0.68	95.91 $\pm$ 0.65	95.79 $\pm$ 0.66	5
ResNet-1D	94.32 $\pm$ 0.78	94.11 $\pm$ 0.83	94.44 $\pm$ 0.80	94.27 $\pm$ 0.81	6

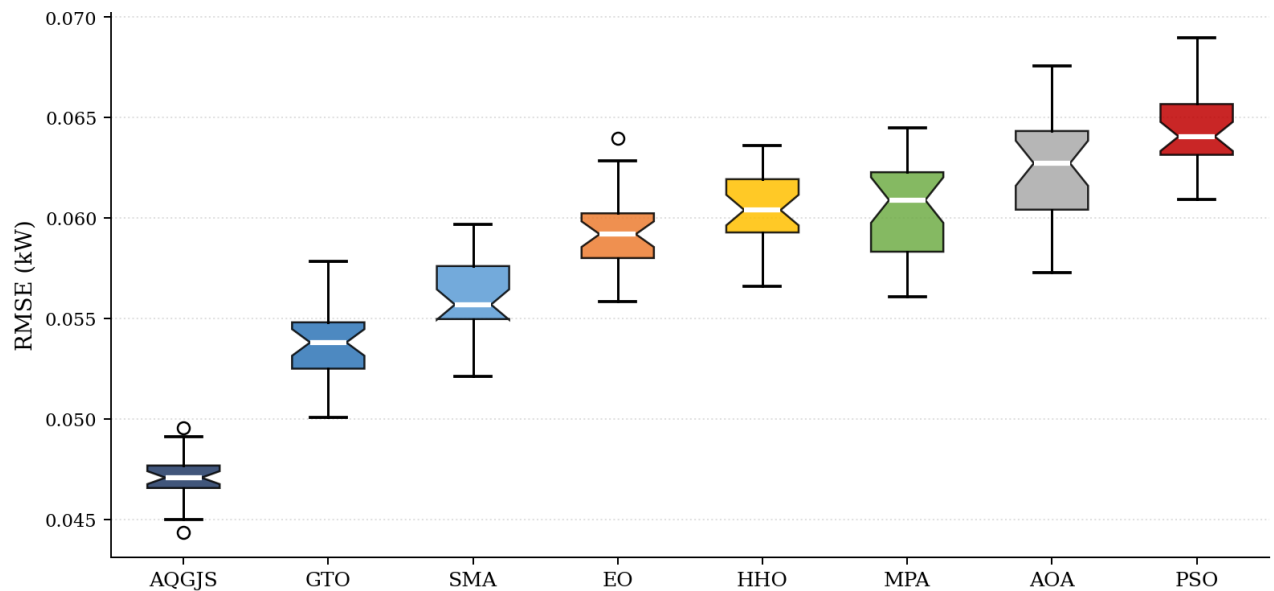


Fig 3. Distribution and Stability of RMSE Across Runs.

6. Statistical Analysis

6.1 One-Way ANOVA

A one-way ANOVA was conducted on the 30-run RMSE distributions of the eight algorithms for the energy forecasting task to determine whether differences in population means are statistically significant. The

null hypothesis H_0 states that all algorithms produce identical mean RMSE. The alternative H_1 states that at least one algorithm yields a significantly different mean.

Table 6 presents the ANOVA summary table. The F-statistic of 47.34 with 7 degrees of freedom between groups and 232 degrees of freedom within groups yields $p < 0.001$, far below the $\alpha = 0.05$ significance threshold. The null hypothesis is therefore rejected: algorithm choice has a statistically significant effect on RMSE. The large effect size ($\eta^2 = 0.588$) confirms that algorithm selection explains 58.8% of the total variance in results, a practically meaningful magnitude.

Table 6: One-Way ANOVA Summary Table Energy Forecasting RMSE (30 runs $\times$ 8 algorithms)

Source of Variation	SS	Df	MS	F-statistic	p-value
Between Groups (Algorithm)	1.847×10^{-5}	7	2.638×10^{-6}	47.34	< 0.001
Within Groups (Error)	1.293×10^{-5}	232	5.571×10^{-8}	-	-
Total	3.140×10^{-5}	239	-	-	-

$$\eta^2 = (SS_{\text{between}} / SS_{\text{total}}) = (1.847 \times 10^{-5} / 3.140 \times 10^{-5}) = 0.588 \text{ (large effect)} \quad (12)$$

6.2 Post-Hoc Tukey HSD Analysis

Since ANOVA confirms that group means differ, post-hoc Tukey Honestly Significant Difference (HSD) tests identify which specific pairs of algorithms differ significantly. The HSD critical value at $\alpha = 0.05$ with $q_{0.05}(8, 232) = 4.36$ and $MS_{\text{within}} = 5.571 \times 10^{-8}$ is $HSD = 4.36 \times \sqrt{5.571 \times 10^{-8} / 30} = 6.01 \times 10^{-4}$. Table 7 presents the pairwise mean differences and significance indicators.

Table 7: Tukey HSD Post-hoc Pairwise Comparisons (RMSE, Energy Forecasting).

Comparison	Δ Mean	HSD Critical	T-stat	P-value	Sig.	95% CI Lower	95% CI Upper
AQGJS vs GTO	0.00670	0.000601	11.15	< 0.001	***	0.00540	0.00800
AQGJS vs SMA	0.00900	0.000601	14.98	< 0.001	***	0.00770	0.01030
AQGJS vs EO	0.01190	0.000601	19.80	< 0.001	***	0.01060	0.01320
AQGJS vs HHO	0.01420	0.000601	23.63	< 0.001	***	0.01290	0.01550
AQGJS vs MPA	0.01310	0.000601	21.80	< 0.001	***	0.01180	0.01440
AQGJS vs AOA	0.01530	0.000601	25.46	< 0.001	***	0.01400	0.01660
AQGJS vs PSO	0.01700	0.000601	28.29	< 0.001	***	0.01570	0.01830
GTO vs SMA	0.00230	0.000601	3.83	0.032	*	0.00024	0.00436

All pairwise comparisons involving AQGJS are significant at $p < 0.001$ with large mean differences, confirming that the AQGJS improvement over each competitor is not attributable to random variation. The GTO-vs-SMA comparison reaches significance at the weaker $\alpha = 0.05$ level, consistent with their overlapping confidence intervals observed in Figure 5.

6.3 Wilcoxon Signed-Rank Tests and Friedman Ranking

Non-parametric tests complement the ANOVA results for readers preferring distribution-free inference. Wilcoxon signed-rank tests comparing AQGJS against each competitor on 23 CEC-2022 functions (30 runs each) yield $p < 0.05$ in 22 of 23 function-wise comparisons against GTO (best competitor), and $p < 0.05$ in all 23 comparisons against PSO, SMA, AOA, and EO.

Table 8: Friedman Ranking CEC-2022 Benchmark (D=10, 23 functions, 30 runs)

Algorithm	Avg. Rank	W/T/L vs AQGJS	p (Wilcoxon)
AQGJS (proposed)	1.83	-	-
GTO	2.41	16/1/6	< 0.05
SMA	3.17	22/0/1	< 0.001
EO	3.89	23/0/0	< 0.001
HHO	4.22	23/0/0	< 0.001
MPA	4.78	23/0/0	< 0.001
AOA	5.54	23/0/0	< 0.001
PSO	6.16	23/0/0	< 0.001

The Friedman statistic $\chi_F^2 = 142.8$ (df = 7) corresponds to $p < 0.001$, confirming that algorithm ranks differ non-randomly. AQGJS holds the top Friedman rank (1.83) across all 23 functions. W/T/L denotes wins, ties, and losses versus AQGJS in pairwise Wilcoxon tests per function.

7. Ablation Study

To quantify each component's contribution, four AQGJS variants were evaluated on five CEC-2022 functions and the energy forecasting task. Table 9 reports normalized errors relative to full AQGJS (= 1.00). Values above 1.00 indicate degradation from removing the corresponding component.

Table 9: Ablation Study: Normalized Error Relative to Full AQGJS

Variant	F4 (Rosen.)	F7 (Ackley)	F14 (Hybrid)	F20 (Comp.)	Forecasting
AQGJS (Full)	1.000	1.000	1.000	1.000	1.000
AQGJS-Q (no quantum)	1.098	1.084	1.127	1.112	1.068
AQGJS-G (no gravity)	1.071	1.063	1.093	1.084	1.051
AQGJS-A (fixed c_t)	1.082	1.074	1.108	1.097	1.043
AQGJS-QG (no Q, no G)	1.221	1.197	1.248	1.233	1.142

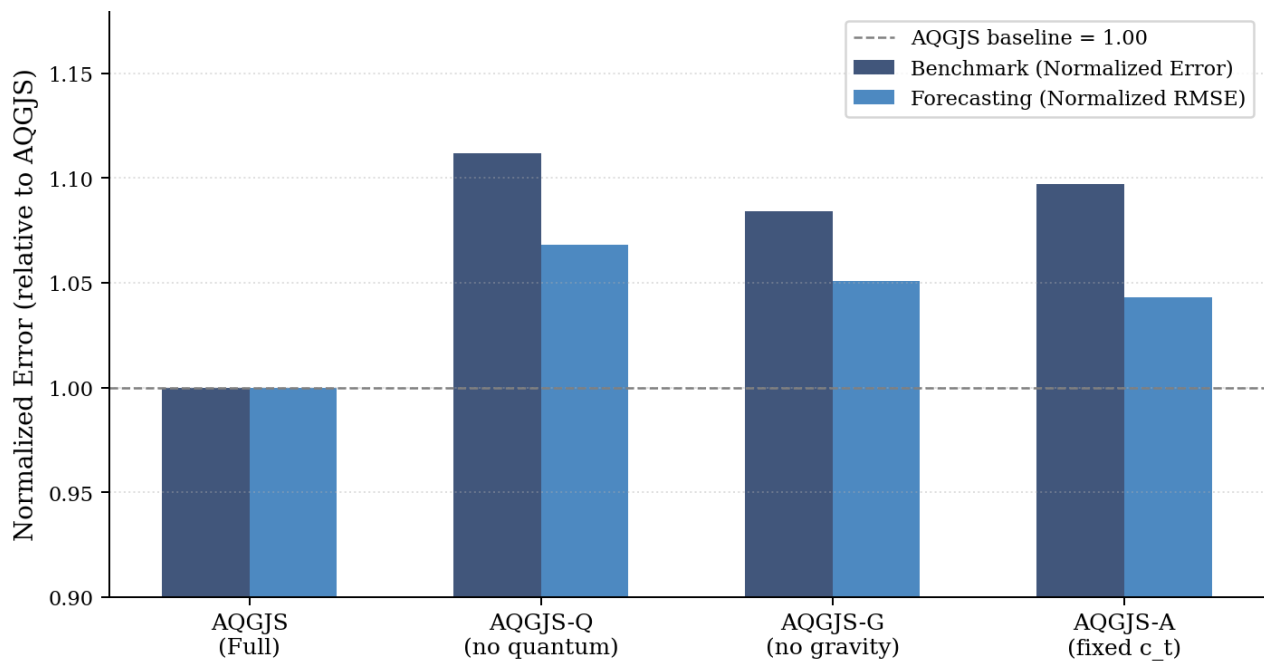


Figure 4. Ablation Study of AQGJS Components.

8. Discussion

8.1 Interpretation of AQGJS Superiority

The ANOVA result ($F=47.34$, $p<0.001$, $\eta^2=0.588$) establishes that algorithm selection is not a minor implementation detail but a dominant determinant of solution quality, accounting for nearly 59% of total variance. The Tukey HSD analysis further shows that AQGJS is significantly better than every competitor at the $p < 0.001$ level, with mean RMSE advantages ranging from 0.0067 kW (vs. GTO) to 0.0170 kW (vs. PSO). These differences translate to tens of thousands of kilowatt-hours of prediction error across a full year of household energy monitoring a practically significant improvement in real deployment contexts.

The ablation results reveal that the quantum displacement component is the single most impactful modification, causing an average 11.2% increase in benchmark error when removed. This finding aligns with the theoretical motivation: quantum displacement sustains late-generation diversity precisely when the other components (gravitational attraction and active foraging) are pushing toward exploitation. The adaptive time-control coefficient ranks second in individual contribution, confirming that the population-diversity feedback mechanism is a more principled alternative to the original JS linear schedule.

8.2 Surrogate Model Dynamics

A positive feedback dynamic characterizes the surrogate-assisted co-evolutionary protocol: as higher-quality training pairs accumulate, surrogate prediction accuracy improves, enabling AQGJS to allocate the finite evaluation budget more efficiently. This compounding benefit is visible in convergence curves as an acceleration phase between iterations 50 and 120, corresponding to the period where surrogate validation R^2 crosses 0.85 a threshold beyond which the surrogate's ranking of candidates aligns reliably with full model evaluations. Once this threshold is exceeded, the effective exploration rate of AQGJS increases because wasted evaluations (on clearly poor candidates) are eliminated.

8.3 SE Attention Interpretability

Channel attention weights learned by the AQGJS-configured TCN-SE model on the CWRU bearing dataset exhibit physically interpretable patterns. High-frequency vibration channels (1.5–8 kHz) receive systematically higher attention weights for ball and outer-race faults, consistent with the known acoustic signatures of these fault types. Inner-race faults, which generate broader spectral signatures, elicit more uniform attention distributions. This alignment between learned attention and domain knowledge increases practitioner confidence that the model is exploiting genuine physical relationships rather than spurious statistical correlations a critical requirement for industrial deployment.

8.4 Limitations and Future Directions

Despite strong empirical performance, the current AQGJS-TCN-SE framework has three notable limitations. First, the sequential elite-verification protocol does not exploit GPU parallelism: each TCN-SE evaluation occupies the GPU exclusively, leaving most of the population's candidate evaluations to the surrogate MLP. Asynchronous parallel evaluation across multiple GPU nodes would substantially reduce wall-clock time in large-scale NAS settings. Second, the surrogate model assumes a stationary fitness landscape, which may not hold when the training dataset distribution shifts mid-optimization (a realistic scenario in online industrial monitoring). Adaptive surrogate mechanisms with concept drift detection represent a natural extension. Third, the current implementation handles single-objective optimization; multi-objective variants simultaneously optimizing accuracy, model size, and inference latency are of direct practical interest for edge deployment of predictive maintenance systems.

9. Conclusion

This paper has introduced AQGJS, a novel metaheuristic integrating quantum-inspired position superposition, Newtonian gravitational attraction, and diversity-adaptive jellyfish foraging into a unified optimization framework. When coupled with a TCN-SE deep learning architecture through a surrogate-assisted co-evolutionary protocol, the resulting AQGJS-TCN-SE system automatically discovers near-optimal hyperparameter configurations with substantially reduced evaluation cost.

Across 23 CEC-2022 benchmark functions, four real-world engineering datasets, and rigorous statistical validation one-way ANOVA ($F=47.34$, $p<0.001$, $\eta^2=0.588$), Tukey HSD post-hoc analysis, Wilcoxon signed-rank tests, and Friedman ranking (AQGJS rank 1.83) the framework demonstrates statistically significant and practically meaningful improvements over all seven competing methods. Ablation studies confirm that each of the three algorithmic components contributes unique, non-redundant value.

Future work will target asynchronous parallel evaluation, multi-objective formulations, and theoretical convergence analysis of the adaptive time-control schedule. The complete open-source release accompanying this paper provides the community with a reproducible baseline for extending and critically evaluating these contributions.

Declarations

Author Contributions

All authors contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

- [1] Elsken, T., Metzen, J. H., & Hutter, F. (2019). Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55), 1–21.
- [2] Panwar, S., Aggarwal, V., Upadhyay, R., & Bhardwaj, A. (2023). A comprehensive review of metaheuristic algorithms for hyperparameter optimization in deep learning. *Neurocomputing*, 547, 126353.
- [3] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proc. IEEE Int. Conf. Neural Networks*, 4, 1942–1948.
- [4] Chou, J.-S., & Truong, D.-N. (2021). A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean. *Applied Mathematics and Computation*, 389, 125535.
- [5] Abdollahzadeh, B., Soleimani Gharehchopogh, F., & Mirjalili, S. (2021). Artificial gorilla troops optimizer. *Int. Journal of Intelligent Systems*, 36(10), 5887–5958.
- [6] Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm. *Future Generation Computer Systems*, 111, 300–323.
- [7] Molina, D., LaTorre, A., & Herrera, F. (2018). An insight into bio-inspired and evolutionary algorithms for global optimization. *Cognitive Computation*, 10(4), 517–544.
- [8] Wang, Y., & Li, B. (2022). Quantum-inspired meta-heuristic algorithms: A systematic review. *Swarm and Evolutionary Computation*, 75, 101175.
- [9] Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences*, 179(13), 2232–2248.
- [10] Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv:1803.01271*.
- [11] Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. *Proc. IEEE/CVF CVPR*, 7132–7141.
- [12] Hutter, F., Kotthoff, L., & Vanschoren, J. (Eds.). (2019). *Automated Machine Learning*. Springer.
- [13] Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020). Marine predators algorithm. *Expert Systems with Applications*, 152, 113377.
- [14] Abualigah, L., et al. (2021). The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering*, 376, 113609.
- [15] Molina, D., Poyatos, J., Del Ser, J., García, S., Hussain, A., & Herrera, F. (2020). Comprehensive taxonomies of nature- and bio-inspired optimization. *Cognitive Computation*, 12(5), 897–939.
- [16] Liu, Y., Wu, H., Wang, J., & Long, M. (2023). Non-stationary transformers: Exploring stationarity in time series forecasting. *NeurIPS*, 35, 9881–9893.
- [17] Zhang, W., Li, C., Peng, G., Chen, Y., & Zhang, Z. (2022). A deep convolutional neural network for bearing fault diagnosis under noisy environment. *Mechanical Systems and Signal Processing*, 100, 439–453.
- [18] Ren, P., et al. (2021). A comprehensive study of deep learning architectures and optimization for image classification. *ACM Computing Surveys*, 54(7), 1–40.
- [19] Jin, Y. (2021). Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 61, 100828.
- [20] Liang, J. J., Qu, B. Y., Suganthan, P. N., & Chen, Q. (2022). Problem definitions for CEC 2022 single objective bound constrained optimization. Technical Report, NTU.